**Controller**

**Consulta Inner join**

var innerJoin = from s in context.Servico join a in context.Agendamento on s.Id equals a.Id where a.Id == agendamento.Id

select new { valor = s.Preco};

from s in db.Services

join sa in db.ServiceAssignments on s.Id equals sa.ServiceId

where sa.LocationId == 1

select s

**Feito:**

**Usuario** (Ok)

**Tipo Contato** (Ok)

**Regime Contratual** (OK)

**Genero** (Ok)

**Cargo** (Ok)

* Trocar mapeamento de serviço para cargo (Um serviço somente pode estar para um cargo e um cargo poderá ter vários serviços)
* Criar o mapeamento

**Servico** (Ok)

* Tirar mapeamento de colaborador para serviço

**Colaborador (Ok)**

- Verificar as propriedades do colaborador.

- Paginar (Adicionar os contatos e tipos).

- Selecionar (Adicionar os contatos e tipos)

- Adicionar o contato com o id do colaborador que é o mesmo da pessoa.(Ok)

- Não colocar deletar em colaborador

**Contrato Trabalho (Ok)**

- Colocar a propriedade porcentagem: decimal (Criado)

- Adicionado o salário como propriedade.

- Não foi adicionado o end-point Deletar.

- Os dois put estão funcionando.

-Get funcionando.

**Contato** (Ok)

- Atualizar (Adicionar os contatos e tipos)

- Paginar os contatos dos clientes.

- Paginar os contatos dos colaboradores.

- Selecionar contatos de um colaborador.

- Selecionar contatos de um cliente.

**Escala** (Ok)

- Incluir

- Paginar por colaborador

- Paginar todos a partir da data atual

- Atualizar

- Deletar

**Cliente** (Ok)

- Criar

- Paginar

- Atualizar

- Deletar em cascata

**Agendamento (Ok)**

- Criar o método para comissão no repositório, que será usado quando o status do agendamento for concluído.

Necessário(id do agendamento)

-Criar o método para modificar o status da escala do colaborador. (Criado e o método também salva a comissão do serviço realizado).

- Atualizar valor ordem serviço (valor total e duração) (Ok)

- Atualizar status da ordem serviço (int statusId)

- Finalizar Agendamento e gera a comissão

- Selecionar por Id (Ok)

- Criado o método Agendar (Ok)

**Ordem Serviço**

- Criar ordem

- Pegar ordem serviço.

- paginar (Ok)

- Encerrar a ordem serviço com o status (OK)

**Comissão**

- Implementar a consulta das comissões

- Implementar a retirada da comissão e lançamento no caixa

**Caixa** (Criado, verificar os métodos e testar)

- Adicionar o relacionamento para a classe Retiradas (Um caixa pode estar para várias retiradas e uma retirada para um caixa)

- Modificar o fechamento de caixa, os valores dos pagamentos serão acrescentados todas as vezes que o pagamento for realizado.

- As retiradas também seguem o mesmo fluxo, serão decrementados do caixa.

**Pagamento** (Não foi criado)

- Pagamento pode ser realizado, sem que a ordem de serviço tenha sido encerrada com status concluído.

- Toda vez que que o pagamento for realizado, já ser acrescentado no caixa. Colocar o usuário que está registrando o pagamento pelo Id.

**Retiradas** (Criar a classe, controller modelos e dtos)

- Será usado para registrar as retiradas diárias

- Conterá a retirada até de salário.

BeneficioController

EspacoCliente (Não foi criado)

EspacoClienteController (Não foi criado)

**Modificações**

GestaoController (Não foi criado) Retirar, não faz sentido (Modificar para financeiro).

**-** Colocar aqui as funções de busca de todos os caixas do mês.

- Buscar o total de pagamentos de salários dos colaboradores e PJ.

- Buscar todas as retiradas, caixas, pagamentos e informar a porcentagem de ganho ou prejuízo.